Network Working Group
Request for Comments: 974

Craig Partridge
CSNET CIC BBN Laboratories Inc
January 1986

MAIL ROUTING AND THE DOMAIN SYSTEM

Status of this Memo

This RFC presents a description of how mail systems on the Internet are expected to route messages based on information from the domain system described in RFCs 882, 883 and 973. Distribution of this memo is unlimited.

Introduction

The purpose of this memo is to explain how mailers are to decide how to route a message addressed to a given Internet domain name. This involves a discussion of how mailers interpret MX RRs, which are used for message routing. Note that this memo makes no statement about how mailers are to deal with MB and MG RRs, which are used for interpreting mailbox names.

Under RFC-882 and RFC-883 certain assumptions about mail addresses have been changed. Up to now, one could usually assume that if a message was addressed to a mailbox, for example, at LOKI.BBN.COM, that one could just open an SMTP connection to LOKI.BBN.COM and pass the message along. This system broke down in certain situations, such as for certain UUCP and CSNET hosts which were not directly attached to the Internet, but these hosts could be handled as special cases in configuration files (for example, most mailers were set up to automatically forward mail addressed to a CSNET host to CSNET-RELAY.ARPA).

Under domains, one cannot simply open a connection to LOKI.BBN.COM, but must instead ask the domain system where messages to LOKI.BBN.COM are to be delivered. And the domain system may direct a mailer to deliver messages to an entirely different host, such as SH.CS.NET. Or, in a more complicated case, the mailer may learn that it has a choice of routes to LOKI.BBN.COM. This memo is essentially a set of guidelines on how mailers should behave in this more complex world.

Readers are expected to be familiar with RFCs 882, 883, and the updates to them (e.g., RFC-973).

What the Domain Servers Know

The domain servers store information as a series of resource records (RRs), each of which contains a particular piece of information about a given domain name (which is usually, but not always, a host). The simplest way to think of a RR is as a typed pair of datum, a domain name matched with relevant data, and stored with some additional type information to help systems determine when the RR is relevant. For the purposes of message routing, the system stores RRs known as MX RRs. Each MX matches a domain name with two pieces of data, a preference value (an unsigned 16-bit integer), and the name of a host. The preference number is used to indicate in what order the mailer should attempt deliver to the MX hosts, with the lowest numbered MX being the one to try first. Multiple MXs with the same preference are permitted and have the same priority.

In addition to mail information, the servers store certain other types of RR's which mailers may encounter or choose to use. These are: the canonical name (CNAME) RR, which simply states that the domain name queried for is actually an alias for another domain name, which is the proper, or canonical, name; and the Well Known Service (WKS) RR, which stores information about network services (such as SMTP) a given domain name supports.

General Routing Guidelines

Before delving into a detailed discussion of how mailers are expected to do mail routing, it would seem to make sense to give a brief overview of how this memo is approaching the problems that routing poses.

The first major principle is derived from the definition of the preference field in MX records, and is intended to prevent mail looping. If the mailer is on a host which is listed as an MX for the destination host, the mailer may only deliver to an MX which has a lower preference count than its own host.

It is also possible to cause mail looping because routing information is out of date or incomplete. Out of date information is only a problem when domain tables are changed. The changes will not be known to all affected hosts until their resolver caches time out. There is no way to ensure that this will not happen short of requiring mailers and their resolvers to always send their queries to an authoritative server, and never use data stored in a cache. This is an impractical solution, since eliminating resolver caching would make mailing inordinately expensive. What is more, the out-of-date RR problem should not happen if, when a domain table is changed,

affected hosts (those in the list of MXs) have their resolver caches flushed. In other words, given proper precautions, mail looping as a result of domain information should be avoidable, without requiring mailers to query authoritative servers. (The appropriate precaution is to check with a host's administrator before adding that host to a list of MXs).

The incomplete data problem also requires some care when handling domain queries. If the answer section of a query is incomplete critical MX RRs may be left out. This may result in mail looping, or in a message being mistakenly labelled undeliverable. As a result, mailers may only accept responses from the domain system which have complete answer sections. Note that this entire problem can be avoided by only using virtual circuits for queries, but since this situation is likely to be very rare and datagrams are the preferred way to interact with the domain system, implementors should probably just ensure that their mailer will repeat a query with virtual circuits should the truncation bit ever be set.

Determining Where to Send a Message

The explanation of how mailers should decide how to route a message is discussed in terms of the problem of a mailer on a host with domain name LOCAL trying to deliver a message addressed to the domain name REMOTE. Both LOCAL and REMOTE are assumed to be syntactically correct domain names. Furthermore, LOCAL is assumed to be the official name for the host on which the mailer resides (i.e., it is not a alias).

Issuing a Query

The first step for the mailer at LOCAL is to issue a query for MX RRs for REMOTE. It is strongly urged that this step be taken every time a mailer attempts to send the message. The hope is that changes in the domain database will rapidly be used by mailers, and thus domain administrators will be able to re-route in-transit messages for defective hosts by simply changing their domain databases.

Certain responses to the query are considered errors:

Getting no response to the query. The domain server the mailer queried never sends anything back. (This is distinct from an answer which contains no answers to the query, which is not an error).

Getting a response in which the truncation field of the header is

set. (Recall discussion of incomplete queries above). Mailers may not use responses of this type, and should repeat the query using virtual circuits instead of datagrams.

Getting a response in which the response code is non-zero.

Mailers are expected to do something reasonable in the face of an error. The behaviour for each type of error is not specified here, but implementors should note that different types of errors should probably be treated differently. For example, a response code of "non-existent domain" should probably cause the message to be returned to the sender as invalid, while a response code of "server failure" should probably cause the message to be retried later.

There is one other special case. If the response contains an answer which is a CNAME RR, it indicates that REMOTE is actually an alias for some other domain name. The query should be repeated with the canonical domain name.

If the response does not contain an error response, and does not contain aliases, its answer section should be a (possibly zero length) list of MX RRs for domain name REMOTE (or REMOTE's true domain name if REMOTE was a alias). The next section describes how this list is interpreted.

Interpreting the List of MX RRs

NOTE: This section only discusses how mailers choose which names to try to deliver a message to, working from a list of RR's. It does not discuss how the mailers actually make delivery. Where ever delivering a message is mentioned, all that is meant is that the mailer should do whatever it needs to do to transfer a message to a remote site, given a domain name for that site. (For example, an SMTP mailer will try to get an address for the domain name, which involves another query to the domain system, and then, if it gets an address, connect to the SMTP TCP port). The mechanics of actually transferring the message over the network to the address associated with a given domain name is not within the scope of this memo.

It is possible that the list of MXs in the response to the query will be empty. This is a special case. If the list is empty, mailers should treat it as if it contained one RR, an MX RR with a preference value of 0, and a host name of REMOTE. (I.e., REMOTE is its only MX). In addition, the mailer should do no further processing on the list, but should attempt to deliver the message to REMOTE. The idea

here is that if a domain fails to advertise any information about a particular name we will give it the benefit of the doubt and attempt delivery.

If the list is not empty, the mailer should remove irrelevant RR's from the list according to the following steps. Note that the order is significant.

For each MX, a WKS query should be issued to see if the domain name listed actually supports the mail service desired. MX RRs which list domain names which do not support the service should be discarded. This step is optional, but strongly encouraged.

If the domain name LOCAL is listed as an MX RR, all MX RRs with a preference value greater than or equal to that of LOCAL's must be discarded.

After removing irrelevant RRs, the list can again be empty. This is now an error condition and can occur in several ways. The simplest case is that the WKS queries have discovered that none of the hosts listed supports the mail service desired. The message is thus deemed undeliverable, though extremely persistent mail systems might want to try a delivery to REMOTE's address (if it exists) before returning the message. Another, more dangerous, possibility is that the domain system believes that LOCAL is handling message for REMOTE, but the mailer on LOCAL is not set up to handle mail for REMOTE. For example, if the domain system lists LOCAL as the only MX for REMOTE, LOCAL will delete all the entries in the list. But LOCAL is presumably querying the domain system because it didn't know what to do with a message addressed to REMOTE. Clearly something is wrong. How a mailer chooses to handle these situations is to some extent implementation dependent, and is thus left to the implementor's discretion.

If the list of MX RRs is not empty, the mailer should try to deliver the message to the MXs in order (lowest preference value tried first). The mailer is required to attempt delivery to the lowest valued MX. Implementors are encouraged to write mailers so that they try the MXs in order until one of the MXs accepts the message, or all the MXs have been tried. A somewhat less demanding system, in which a fixed number of MXs is tried, is also reasonable. Note that multiple MXs may have the same preference value. In this case, all MXs at with a given value must be tried before any of a higher value are tried. In addition, in the special case in which there are several MXs with the lowest preference value, all of them should be tried before a message is deemed undeliverable.

Minor Special Issues

There are a couple of special issues left out of the preceding section because they complicated the discussion. They are treated here in no particular order.

Wildcard names, those containing the character '*' in them, may be used for mail routing. There are likely to be servers on the network which simply state that any mail to a domain is to be routed through a relay. For example, at the time that this RFC is being written, all mail to hosts in the domain IL is routed through RELAY.CS.NET. This is done by creating a wildcard RR, which states that *.IL has an MX of RELAY.CS.NET. This should be transparent to the mailer since the domain servers will hide this wildcard match. (If it matches *.IL with HUJI.IL for example, a domain server will return an RR containing HUJI.IL, not *.IL). If by some accident a mailer receives an RR with a wildcard domain name in its name or data section it should discard the RR.

Note that the algorithm to delete irrelevant RRs breaks if LOCAL has a alias and the alias is listed in the MX records for REMOTE. (E.g. REMOTE has an MX of ALIAS, where ALIAS has a CNAME of LOCAL). This can be avoided if aliases are never used in the data section of MX RRs.

Implementors should understand that the query and interpretation of the query is only performed for REMOTE. It is not repeated for the MX RRs listed for REMOTE. You cannot try to support more extravagant mail routing by building a chain of MXs. (E.g. UNIX.BBN.COM is an MX for RELAY.CS.NET and RELAY.CS.NET is an MX for all the hosts in .IL, but this does not mean that UNIX.BBN.COM accepts any responsibility for mail for .IL).

Finally, it should be noted that this is a standard for routing on the Internet. Mailers serving hosts which lie on multiple networks will presumably have to make some decisions about which network to route through. This decision making is outside the scope of this memo, although mailers may well use the domain system to help them decide. However, once a mailer decides to deliver a message via the Internet it must apply these rules to route the message.

Examples

To illustrate the discussion above, here are three examples of how mailers should route messages. All examples work with the following database:

```
A.EXAMPLE.ORG     IN     MX     10     A.EXAMPLE.ORG
A.EXAMPLE.ORG     IN     MX     15     B.EXAMPLE.ORG
A.EXAMPLE.ORG     IN     MX     20     C.EXAMPLE.ORG
A.EXAMPLE.ORG     IN     WKS    10.0.0.1     TCP     SMTP

B.EXAMPLE.ORG     IN     MX     0       B.EXAMPLE.ORG
B.EXAMPLE.ORG     IN     MX     10      C.EXAMPLE.ORG
B.EXAMPLE.ORG     IN     WKS    10.0.0.2     TCP     SMTP

C.EXAMPLE.ORG     IN     MX     0      C.EXAMPLE.ORG
C.EXAMPLE.ORG     IN     WKS    10.0.0.3     TCP     SMTP

D.EXAMPLE.ORG     IN     MX     0      D.EXAMPLE.ORG
D.EXAMPLE.ORG     IN     MX     0      C.EXAMPLE.ORG
D.EXAMPLE.ORG     IN     WKS    10.0.0.4     TCP     SMTP
```

In the first example, an SMTP mailer on D.EXAMPLE.ORG is trying to deliver a message addressed to A.EXAMPLE.ORG. From the answer to its query, it learns that A.EXAMPLE.ORG has three MX RRs. D.EXAMPLE.ORG is not one of the MX RRs and all three MXs support SMTP mail (determined from the WKS entries), so none of the MXs are eliminated. The mailer is obliged to try to deliver to A.EXAMPLE.ORG as the lowest valued MX. If it cannot reach A.EXAMPLE.ORG it can (but is not required to) try B.EXAMPLE.ORG. and if B.EXAMPLE.ORG is not responding, it can try C.EXAMPLE.ORG.

In the second example, the mailer is on B.EXAMPLE.ORG, and is again trying to deliver a message addressed to A.EXAMPLE.ORG. There are once again three MX RRs for A.EXAMPLE.ORG, but in this case the mailer must discard the RRs for itself and C.EXAMPLE.ORG (because the MX RR for C.EXAMPLE.ORG has a higher preference value than the RR for B.EXAMPLE.ORG). It is left only with the RR for A.EXAMPLE.ORG, and can only try delivery to A.EXAMPLE.ORG.

In the third example, consider a mailer on A.EXAMPLE.ORG trying to deliver a message to D.EXAMPLE.ORG. In this case there are only two MX RRs, both with the same preference value. Either MX will accept messages for D.EXAMPLE.ORG. The mailer should try one MX first (which one is up to the mailer, though D.EXAMPLE.ORG seems most reasonable), and if that delivery fails should try the other MX (e.g. C.EXAMPLE.ORG).